
Ruby on Rails

— Integración Rails con HTTParty —

Overview

- Interacción entre Rails y httparty

Agregar al Gemfile

```
Gemfile x
31 # bundle exec rake doc:rails generates the API under d
32 gem 'sdoc', require: false
33 end
34
35 # Use ActiveModel has_secure_password
36 # gem 'bcrypt', '~> 3.1.7'
37
38 # Use unicorn as the app server
39 # gem 'unicorn'
40
41 # Use Capistrano for deployment
42 # gem 'capistrano', group: :development
43
44 # Use debugger
45 # gem 'debugger', group: [:development, :test]
46 gem 'httparty', '0.14.0'
47
```

Ejecutar bundle

```
→ my_first_app git:(master) * bundle install
Resolving dependencies...
Rubygems 2.0.14 is not threadsafe, so your gems may be
Using rake 11.3.0
Using i18n 0.7.0
Using minitest 4.7.5
Using multi_json 1.12.1
Using thread_safe 0.3.5
Using tzinfo 0.3.52
Using builder 3.1.4
Using erubis 2.7.0
Using rack 1.5.5
Using mime-types 1.25.1
Using polyglot 0.3.5
Using activerecord-deprecated_finders 1.0.4
Using arel 4.0.2
Using bundler 1.13.5
Using coffee-script-source 1.10.0
Using execjs 2.7.0
Using thor 0.19.1
Using hike 1.2.3
Using multi_xml 0.5.5
Using json 1.8.3
Using tilt 1.4.1
```

Cada vez que se ejecuta el bundler es importante reiniciar el servidor.

Mostremos la traducción de un texto

- Mostremos la traducción de un texto
- Generemos el Controller: Data

Crear controller

`rails g controller translators translate detect languages`

```
MyFirstApp::Application.routes.draw do
  get "translators/translate"
  get "translators/detect"
```

Generated sources

```
MyFirstApp::Application.routes.draw do
  get "translators/translate"
  get "translators/detect"
  get "translators/languages"
```

```
class TranslatorsController < ApplicationController

  def translate
  end

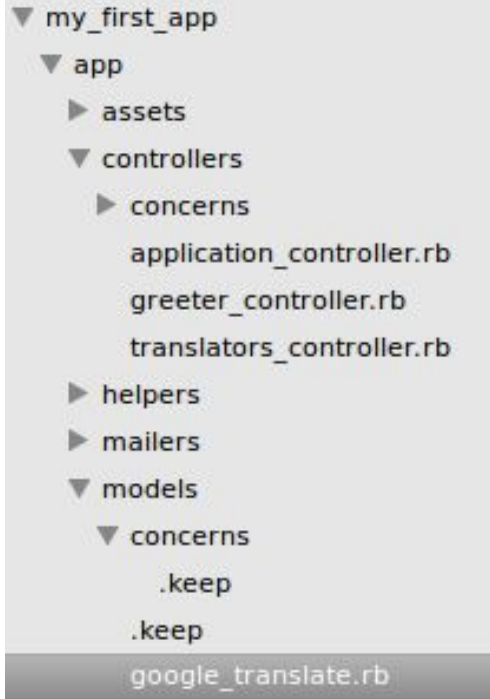
  def detect
  end

  def languages
  end

end
```

```
▼ views
  ► greeter
  ► layouts
  ▼ translators
    detect.html.erb
    languages.html.erb
    translate.html.erb
```

Crear un modelo (de acuerdo al servicio)



```
google_translate.rb x translators_controller.rb x
1 class GoogleTranslate
2   include HTTParty
3   default_options.update(verify: false)
4   base_uri "https://www.googleapis.com/language/translate/v2"
5   format :json
6
7   @@api_key = "AIzaSyBdqIJZunlGxiU5ubdmKlp2VqxqR7GYLsQ"
8
9   default_params key: @@api_key
10
11   attr_accessor :source, :target
12
13   def initialize()
14     end
15
16   def initialize(source, target)
17     @source = source
18     @target = target
19   end
end
```


Controllers -> Plural / Models -> Singular

Implementación del modelo...

```
class GoogleTranslate
  include HTTParty
  default_options.update(verify: false)
  base_uri "https://www.googleapis.com/language/translate/v2"
  format :json

  @@api_key = "AIzaSyBdqijJZunlGxiU5ubdmKlp2VqxqR7GYLsQ"

  default_params key: @@api_key

  attr_accessor :source, :target

  def initialize(source, target)
    @source = source
    @target = target
  end
end
```

Implementación del modelo

```
def translate(search)
  options = { query: {q: search, source:@source, target:@target}}
  response = self.class.get("", options)
  if response.success?
    response
  else
    raise response.response
  end
end

def self.detect(target_language)
  options = { query: {q: target_language}}
  response = get("/detect", options)
  if response.success?
    response
  else
    raise response.response
  end
end |
```

Ejercicio:

Crear un método de clase que invoque al servicio `api/languages` y obtenga una **lista de lenguajes** con los que puede interactuar un determinado **target_language**.

Métodos del controller

```
class TranslatorsController < ApplicationController
  def translate
    @google = GoogleTranslate.new("en", "es")
    @trans = "This is a really good party!"
    @response = @google.translate(@trans)
    @translations = @response['data']['translations'][0]['translatedText']
  end

  def detect
    @trans = "This is a really good party!"
    @detection = GoogleTranslate.detect(@trans)
    puts "#{@detection}"
  end

  def languages
    @target_language = "fr"
    @languages = GoogleTranslate.languages(@target_language)
    puts "Es posible traducir desde los siguientes idiomas: #{@list}"
  end
end
```

Vista: Languages

```
<h1>Detectando idiomas que pueden interactuar con el Francés - <%= @target_language %></h1>
```

```
<table border="1">
  <tr>
    <th>Language</th>
    <th>Name</th>
  </tr>
  <% @languages['data']['languages'].each do |lang| %>
    <tr>
      <td><%= lang['name'] %></td>
      <td><%= lang['language'] %></td>
    </tr>
  <% end %>
</table>
```

Vista: Translate

```
<h1>Translated text!</h1>  
<p><%= @translations%></p>
```

Vista: Detect

```
<h1>Detected language !</h1>  
<h2><%= @detection['data']['detections'][0][0]['language'] %></h2>
```

Entonces...

- Por convención los **controladores** están definidos en **plural**
- Por convención los **modelos** están definidos en **singular**
- Los archivos en el directorio **app/** están automáticamente incluidos.